U.S. NAVAL ACADEMY
COMPUTER SCIENCE DEPARTMENT
TECHNICAL REPORT

The McCallum Projection, Lifting, and Order-Invariance

Brown, Christopher W.

USNA-CS-TR-2005-02

May 3, 2005

USNA Computer Science Dept. ◦ 572M Holloway Rd Stop 9F ◦ Annapolis, MD 21403

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
22 APR 2005		**00-04-2005 to 00-04-2005**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
The McCallum projection, lifting, and order-invariance	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **U.S. Naval Academy,Computer Science Department,572M Holloway Rd Stop 9F,Annapolis,MD,21403**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**15**	

The McCallum projection, lifting, and order-invariance*

Christopher W. Brown
Computer Science Department, Stop 9F
United States Naval Academy
572M Holloway Road
Annapolis, MD 21402
wcbrown@usna.edu

April 22, 2005

Abstract

The McCallum Projection for Cylindrical Algebraic Decomposition (CAD) produces a smaller projection factor set than previous projections, however it does not always produce a sign-invariant CAD for the set of input polynomials. Problems may arise when a $(k+1)$-level projection factor vanishes identically over a k-level cell. According to McCallum's paper, when this happens (and $k+1$ is not the highest level in the CAD) we do not know whether the projection is valid, i.e. whether or not a sign-invariant CAD for the set of input polynomials will be produced when lifting is performed in the usual way. When the k-level cell in question has dimension 0, McCallum suggests a modification of the lifting method that will ensure the validity of his projection, although to my knowledge this has never been implemented.

In this paper we give easily computable criteria that often allow us to conclude that McCallum's projection is valid even though a projection factor vanishes identically over a cell. We also improve on McCallum's modified lifting method.

We've incorporated the ideas contained in this paper into QEPCAD, the most complete implementation of CAD. When McCallum's projection is invalid because of a projection factor not being order-invariant over a region on which it vanishes identically, at least a warning message

*This article was originally posted on my website in September 2001 as `MOTS2001.1.ps.gz`. It is now being reformatted and republished as a USNA CS Department Technical Report. All references to "current" are to be taken as "current as of 1 September, 2001.

ought to be issued. Currently, QEPCAD may print warning messages that are not needed, and may fail to print warning messages when they are needed. Our implementation in QEPCAD ensures that warning messages are printed when needed, and reduces the number of times warning messages are printed when not needed. Neither McCallum's modified lifting method nor our improvement of it have been implemented in QEPCAD — the design of the system would make implementing such a feature quite difficult.

1 Introduction

The McCallum Projection [11] (and the Improved McCallum Projection [3]) represents a huge improvement over the original projection [6] for CAD construction, as the projection factor set it produces is much smaller. Unfortunately, it also involves a slightly more complicated idea of lifting. In particular, lifting becomes difficult when a projection factor vanishes identically over some set (for example, as $(x+y)z-(x-y)$ vanishes identically over $(x=0, y=0)$). According to McCallum's original paper, when a projection factor vanishes identically over a region of dimension greater than zero, the projection fails (unless that projection factor is of the highest level). If a projection factor vanishes identically over a region of dimension zero, then a "delineating polynomial" must be added to the lift basis in that step (once again, unless that projection factor is of the highest level). The key idea is that iterating the lifting process requires not simply *sign-invariance*, but the stronger property of *order-invariance*, and while a polynomial is sign-invariant over a region over which it is identically zero, it is not necessarily order-invariant.

In this paper, we describe more precise (and yet still easily computable) criteria for determining when order-invariance is required, when a delineating polynomial needs to be added to ensure order-invariance, and what that delineating polynomial should be. These improvements are of considerable practical importance. When the McCallum projection fails, either direct and substantial work by hand is required of the user, or the McCallum Projection needs to be abandoned in favor of far costlier projections.

We consider in this paper the QEPCAD system, which provides the most complete implementation of CAD. When the McCallum Projection requires the addition of a delineating polynomial, QEPCAD fails! [1] Not only doesn't QEPCAD currently include facilities for adding delineating polynomials, but its architecture makes it extremely difficult to add such a feature. Thus, apart from

[1] In fact, the current situation is worse: QEPCAD warns the user if a projection factor not of the highest level vanishes identically over a positive dimensional cell, but simply **ignores** cases in which a projection factor vanishes over a 0-dimensional cell! It may be possible that QEPCAD gives incorrect answers as a result.

any performance benefit of not adding unnecessary delineating polynomials, identifying only those cases in which they are truly necessary is critical.

These improvements have been implemented in QEPCAD, with the result that for the first time the user is able to use the McCallum Projection and be assured that, in the absence of error messages (such as "A Delineating Polynomial Must Be Added"), the McCallum Projection is definitely valid. When a projection factor P vanishes identically over cell c:

1. We determine whether order-invariance is really required for P (described in Section 4), if not QEPCAD simply continues.
2. When c is 0-dimensional we determine whether a delineating polynomial must be added to ensure the order-invariance of P over c (described in Section 2), if so, we print an error message, otherwise QEPCAD simply continues.
3. When c has positive dimension, we are sometimes able to determine that no delineating polynomial is needed to ensure the order-invariance of P over c (described in Section 3) and in this case QEPCAD simply continues — otherwise we print an error message.

This paper assumes that the reader is familiar with the McCallum projection and the basic methods of CAD and quantifier elimination by CAD as explained in, for example, [7] or [2].

2 The zero-dimensional case

Suppose that $P(x_1, \ldots, x_{k+1})$ is a projection factor that vanishes identically at a point $\alpha = (\alpha_1, \ldots, \alpha_k)$, i.e. $P(\alpha_1, \ldots, \alpha_k, x_{k+1}) = 0$. In this section we consider the problem of constructing a decomposition of the line $\alpha \times \mathbb{R}$ into regions in which the order of P is constant.

2.1 Minimal delineating polynomials

When projection factor $P(x_1, \ldots, x_{k+1})$ vanishes identically over a point $\alpha \in \mathbb{R}^k$, we know that P has order greater than 0 in $\alpha \times \mathbb{R}$, but we don't know whether or not the order of P is constant in that line. Suppose t is the smallest index for which at least one of the t-order partials of P is not identically zero over α, and let D_P be the set of all t-order partials of P. We know that the order of P in $\alpha \times \mathbb{R}$ is t almost everywhere, but at finitely many points the order may

be greater than t. In fact, the order of P in $\alpha \times \mathbb{R}$ is greater than t at exactly those points at which all the elements of D_P are zero.

We would like to introduce a "delineating polynomial" to define our decomposition of $\alpha \times \mathbb{R}$. A delineating polynomial is a nonzero polynomial in $\mathbb{R}[x_{k+1}]$ whose roots include the x_{k+1}-coordinate of every point in $\alpha \times \mathbb{R}$ at which the order of P is greater than t. McCallum points out that any element of D_P that is not identically zero over α may be used as a delineating polynomial. However, while the vanishing of such a polynomial is a *necessary* condition for an increase in the order of P, it is not a *sufficient* condition. *All* elements of D_P must vanish at a point in $\alpha \times \mathbb{R}$ for the order to increase. A polynomial which vanishes at exactly the the x_{k+1}-coordinate of every point in $\alpha \times \mathbb{R}$ at which the order of P is greater than t will be called a *minimal delineating polynomial*. The vanishing of a minimal delineating polynomial provides a necessary and sufficient condition for the increase in order of P.

Constructing a minimal delineating polynomial is quite easy. If we define

$$S = \{Q(\alpha_1, \ldots, \alpha_k, x_{k+1}) \mid Q \in D_P\}$$

the order of P is greater than t at exactly the points of the form $(\alpha_1, \ldots, \alpha_k, \beta)$, where β is a zero of all the elements of S. If G is the GCD of the elements of S, then the roots of G are exactly the common roots of the elements of S — i.e. G is a minimal delineating polynomial!

Thus, instead of choosing one of the non-zero elements of D_P and using it as a delineating polynomial (as suggested in [11]), we use the GCD of all non-zero elements of D_P as a delineating polynomial.

2.2 In practice

One of the practical benefits of computing a minimal delineating polynomial when faced with a projection factor $P(x_1, \ldots, x_{k+1})$ that vanishes identically over a point $\alpha \in \mathbb{R}^k$ is that one often finds that the minimal delineating polynomial is constant, meaning that no polynomial needs to be added at all! Some examples illustrate this point.

Example 1 Consider the polynomial $P = (x + y - 1)z + (y - 1)$ and the point $\alpha = (0, 1)$. P vanishes identically over α, so according to the McCallum projection we need to add a delineating polynomial to decompose $\alpha \times \mathbb{R}$ into regions in which P is order-invariant.

$$D_P = \{P_x = z, P_y = z + 1, P_z = x + y - 1\}$$

Thus, $S = \{z, z + 1, 0\}$ and G, the GCD of the elements of S, is 1. This means that even though P vanishes identically over α, the order if P is invariant in $\alpha \times \mathbb{R}$. Thus, no delineating polynomial is needed!

Example 2 In the "X Axis Ellipse" problem (see Section 6.1), a well-known problem in the literature, the polynomial

$$P(a,b,c,x) = b^2x^2 - a^2x^2 - 2b^2cx + b^2c^2 - a^2b^2 + a^2$$

appears as a projection factor (it is the resultant of two initial polynomials). The point $\alpha = (1,1,0)$ is a zero-dimensional cell in the CAD constructed for this problem, and P vanishes identically over α.

$$D_P = \left\{ \begin{array}{l} P_a = -2ax^2 - 2ab^2 + 2a, P_b = 2bx^2 - 4bcx + 2bc^2 - 2ba^2, \\ P_c = -2b^2x + 2b^2c, P_x = 2b^2x - 2a^2x - 2b^2c \end{array} \right\}$$

Thus, $S = \{-2x^2, 2x^2 - 2, -2x, 0\}$ and G, the GCD of the elements of S, is 1.Thus, no delineating polynomial is needed!

2.3 Must non-constant minimal delineating polynomials always be added?

From the perspective of QEPCAD, if a delineating polynomial really does need to be added, we are out of luck. Moreover, a considerable amount of work will be required to modify the program to allow the addition of delineating polynomials. So, is all hope lost when we compute a minimal delineating polynomial and it is non-constant? Not always!

Let $P(x_1, \ldots, x_{k+1})$ be our familiar projection factor that vanishes identically over a point $\alpha \in \mathbb{R}^k$. When we lift over α during CAD construction, P is typically one of many $(k+1)$-level projection factors, and we compute a *lift basis*, namely a squarefree basis constructed from substituting α into all $(k+1)$-level projection factors, which defines our decomposition of $\alpha \times \mathbb{R}$. If this lift basis already "contains" the minimal delineating polynomial we've computed (i.e. the squarefree part of the minimal delineating polynomial divides the product of the basis polynomials), then there's nothing to "add"!

This might seem unlikely, but in fact it's not. In Anai's Problem (see Section 6.2), we have a projection factor $P(s,z,x_1,x_2) = sx_2 - x_1^2$. The point $\alpha = (0,-1,0)$ is a zero-dimensional cell in the CAD constructed for this problem, and P vanishes identically over α. $D_P = \{P_s = x_2, P_z = 0, P_{x_1} = -2x_1, P_{x_2} = s\}$, and thus $S = \{x_2, 0, 0, 0\}$ and the minimal delineating polynomial is x_2. From this we see that the order of P in $\alpha \times \mathbb{R}$ is 1 everywhere but $(0,-1,0,0)$, where it is higher (order 2, specifically). It seems that we must add a delineating polynomial. However, x_2 is itself a projection factor, and thus a part of the lift basis. Therefore, the minimal delineating polynomial doesn't need to be added, it's already there! [Note that as a quantifier elimination problem, the vanishing projection factor would not cause difficulties because it is at the highest level, so sign-invariance suffices. What the above argument shows is that what we actually get an order-invariant decomposition.]

3 Dimension greater than zero

In Section 2.1 we deal with the vanishing of projection factor P over a point α by considering the system of all t-order partials of P. We often find that the system is inconsistent, and therefore determine that P is actually order-invariant over α. In this section we try to apply the same ideas to the situation in which P vanishes over a region of higher dimension.

3.1 In theory

If c is a k-level cell in a CAD, it is a connected, open subset of some variety defined by all the projection factors of level k or less that are zero in the cell. In other words, if $f_1, \ldots, f_s$ are the projection factors of level k or less that are zero in c, then c is an open subset of $V(< f_1, \ldots, f_s >)$.

For a $(k+1)$-level projection factor P, the smallest order taken on by P in $c \times \mathbb{R}$ is the smallest index t such that some element of D_P, the set of all t-th order partials of P, is not in $< f_1, \ldots, f_s >$. The question of whether or not P is in fact order invariant over c becomes a decision problem:

$$(\exists x_1) \cdots (\exists x_k)(\exists x_{k+1}) \left[F_c \wedge \left(\bigwedge_{q \in D_P} [q = 0] \right) \right]$$

where F_c is a defining formula for the cell c. This could be solved by CAD, or by other methods. If the formula is true, then P is not order-invariant over c, otherwise P is order invariant over c. In fact, when the above formula is true, this same approach could be used to provide a decomposition of $c \times R$ into regions in which P is order-invariant.

Thus, we can in theory deal with the problem posed by a vanishing projection factor. However, in practice it's not clear how attractive such a solution would be. At the very least one may say that QEPCAD would require a major overhaul to make such things work.

3.2 In practice

In the previous section we saw how to deal with the problem posed by a vanishing projection factor ... in theory, at least. The question we consider here is whether there is a quick and easy test that will, in many practical cases, detect that the polynomial P is in fact order-invariant over positive-dimensional cell c, even though it vanishes identically over c. Another example will illustrate this point.

Example 4 In the "edge-square product" problem (see Section 6.3), we have the projection factor

$$P(x, y, x_1, x_2) = x_1 x_2 - x - 1$$

and the cell $c = \{(-1, y, 0) \in \mathbb{R}^3 \mid y \in (\beta, \gamma)\}$. Clearly, P vanishes identically over c, and c has dimension 1. However, notice that y does not appear in P. In fact, because P depends only on x and x_1, and because the values for x and x_1 are constant within c, this is really just an instance of our zero-dimensional problem — $D_P = \{-1, x_2, x_1\}$, $S = \{-1, x_2, 0\}$, and the GCD of the elements of S is 1. Thus, P has invariant order 1 in $c \times \mathbb{R}$.

In practice, cases in which a projection factor vanishes identically over a region of dimension greater than one are often of this type. The hard part is figuring out whether c is really constant in all the coordinates corresponding to variables that occur in P.

Our approach is to forgo a true decision procedure for determining whether c is really constant in all the coordinates corresponding to variables that occur in P and, instead, settle for a fast procedure that is sometimes able to to prove that c is constant in all the coordinates corresponding to variables that occur in P. Hopefully, the procedure will be able prove that coordinates are constant in many situations in which this problem actually occurs. It is certainly fast, and is able to prove that coordinates are constant for the few uncontrived examples we have found to run it on.

The procedure is called "CONSTCOORDTEST". It takes a level k, a k-level cell c, and a polynomial P in the variables $x_1, \ldots, x_k$. CONSTCOORDTEST returns SUCCESS if it is able to prove that c is constant in all the coordinates corresponding to variables that occur in P, and FAILURE otherwise — which means that it's unable to prove this statement true, not necessarily that the statement is false. Appendix B presents (without proof) a proposal for a true decision procedure for this problem, which points to ways that CONSTCOORDTEST could be strengthened to return SUCCESS in more cases.

CONSTCOORDTEST(k, c, P)

1. set $a = \{\}$ (note: a grows to be sort of a generalized sample point, where some coordinates are given values while others remain free, like $(x = -1, y = y, z = 0)$)
2. for i from 1 to k do
   (a) set c_i to the i-level ancestor of c
   (b) if c_i is a sector then
      i. if $deg_{x_i}(P) = 0$, add $x_i = x_i$ to a, otherwise return FAILURE
   (c) if c_i is a section then

i. set L to the set of i-level projection factors of which c_i is a section (by definition of "section" none of these polynomials vanish identically on c_{i-1})

ii. set L' to the set of all elements of L evaluated at $x_i = \alpha_i$, where α_i is the ith coordinate of the sample point for c_i

iii. set L^* to the set of all elements of L' "evaluated at a" (this is actually a partial evaluation, since some of the entries of a are of the form $x_j = x_j$)

iv. if any element of L^* is the zero polynomial then add $x_i = \alpha_i$ to a, otherwise
if $deg_{x_i}(P) = 0$ then add $x_i = x_i$ to a, otherwise return FAILURE

3. return SUCCESS

To prove that this algorithm is correct, we will show that after i iterations, if the algorithm has not returned FAILURE, all the coordinates that are assigned values in a are "correct" for c_i, in the sense that c_i is constant in those coordinates with those values (although, of course, c_i may be constant in other coordinates as well).

The $i = 0$ case is trivial since $a = \emptyset$.

Suppose $i > 0$, consider the beginning of the ith iteration of the loop Step 2. By induction, a is "correct" for c_{i-1} in the above sense.

c_i is a sector : Step 2.b determines the course of this iteration. If x_i appears in P, FAILURE is returned (as it clearly should be). If x_i does not appear in P, then we leave a unaltered and, because c_i inherits its coordinates for $\{x_1, \ldots, x_{i-1}\}$ from c_{i-1}, a is "correct" for c_i after the ith iteration.

c is a section : Step 2.c(i) sets L to the set of i-level projection factors of which c_i is a section. If f is a continuous real-valued function defined over c_{i-1} whose graph over c_{i-1} intersects c_i in at least one point, then by the usual definitions for CAD:

$$c_i \text{ is the graph of } f \text{ over } c_{i-1} \Longleftrightarrow (\exists p \in L)(\forall \overline{\alpha} \in c_{i-1})[p(\overline{\alpha}, f(\overline{\alpha})) = 0] \tag{1}$$

Now, if $f(x_1, \ldots, x_{i-1}) = \beta$, and β is the ith coordinate of at least one point in c_i, (1) tells us

$$c_i \text{ is the graph of } f = \beta \text{ over } c_{i-1} \Longleftrightarrow (\exists p \in L)(\forall \overline{\alpha} \in c_{i-1})[p(\overline{\alpha}, \beta) = 0] \tag{2}$$

Of course, "c_i is the graph of $f = \beta$ over c_{i-1}" means that c_i has constant ith coordinate with value β. Therefore, we have the following result:

Theorem 1 *Let β is the ith coordinate of at least one point in c_i. Cell c_i has constant ith coordinate if and only if*

$$(\exists p \in L)(\forall \overline{\alpha} \in c_{i-1})[p(\overline{\alpha}, \beta) = 0] \tag{3}$$

Now, our "β" will be α_i, the ith coordinate of the sample point for c_i. Thus, Theorem 1 becomes: c_i has constant ith coordinate if and only if

$$(\exists p \in L)(\forall \overline{\alpha} \in c_{i-1})[p(\overline{\alpha}, \alpha_i) = 0]$$

Step 2.c(ii) sets L' to be the set of all $p \in L$ evaluated at $x_i = \alpha_i$, so we may restate the above as: c_i has constant ith coordinate if and only if

$$(\exists q \in L')(\forall \overline{\alpha} \in c_{i-1})[q(\overline{\alpha}) = 0]$$

Recall that a is a "partial sample point" in which variables are only given values if the associated coordinates in c_{i-1} have been proven by previous loop iterations to be constant. If we let "$|_a$" denote "evaluation at a" in this sense, then $q|_a(\overline{\alpha}) = q(\overline{\alpha})$ for any $\overline{\alpha} \in c_{i-1}$. Step 2.c(iii) computes $L^* = \{q|_a \text{ such that } q \in L'\}$. So the above may be restated as: c_i has constant ith coordinate if and only if

$$(\exists q \in L^*)(\forall \overline{\alpha} \in c_{i-1})[q(\overline{\alpha}) = 0] \tag{4}$$

Now consider the test in Step 2.c(iv). If there is an element q in L^* that is the zero polynomial then clearly $(\forall \overline{\alpha} \in c_{i-1})[q(\overline{\alpha}) = 0]$, which means that (4) is true, which in turn means that c_i has constant ith coordinate (with value α_i). Therefore, if Step 2.c(iv) determines that the coordinate is constant, it is correct. Otherwise, it either returns FAILURE or x_i does not appear in P and it proceeds without assuming that the x_i-coordinate is constant in c_i. Thus, after i iterations, a is "correct" for c_i.

When CONSTCOORDTEST returns success, we find a minimal delineating polynomial for P just as we did in Section 2.1 for the zero-dimensional case. If this minimal delineating polynomial is constant, we know that P is in fact order-invariant in $c \times \mathbb{R}$. Thus, once again, there is no need for QEPCAD to report failure.

4 When is order-invariance really needed

McCallum notes that when our goal is to construct a sign-invariant CAD, as it is for quantifier elimination, order-invariance of projection factors is not required at the highest level. At lower levels we need order-invariance to ensure that subsequent lifting steps are valid, but at the highest level there are no subsequent

lifting steps, and thus sign-invariance is sufficient. A projection factor that is identically zero in some region $c \times \mathbb{R}$ is certainly sign-invariant in the region, and therefore we don't need to worry about delineating polynomials in this case.

In fact whenever the projection factor in question does not have a derivation as a resultant or discriminant of other projection factors only sign-invariance is required. In [3] it is pointed out that only sign-invariance is required for coefficients, and clearly only sign-invariance is required of input polynomials, regardless of their level. In practice many cases in which projection factors vanish identically involve polynomials that have no derivations as resultants or discriminants. This test is an important tool for recognizing situations in which the McCallum projection/lifting method seems to fail, but in fact does not.

5 Conclusion

The McCallum Projection produces much smaller projection factor sets than previous projections, which makes it feasible to attack larger problems than would otherwise have been possible. Unfortunately, CAD construction is complicated by the requirement of sign-invariance rather than simply order-invariance of projection factors in cells. This requirement sometimes causes the McCallum projection to "fail", and sometimes requires the addition of a "delineating polynomial" during lifting.

In theory, adding delineating polynomials does not add significantly to computational costs. In practice, however, it complicates implementation. In the case of QEPCAD, grafting the facilities for adding delineating polynomials would be a significant undertaking. When the McCallum projection fails, we have a bigger problem. Either another projection must be used, which will likely result in a much larger projection factor set, or human intervention will be needed to break the problem up into subproblems for which the McCallum projection can be used.

This paper refines the criteria for determining when delineating polynomials are needed, what delineating polynomial is needed, and when the McCallum projection fails. The result is that we can often safely use the original lifting method with the McCallum projection. Our implementation in QEPCAD of these criteria means that, for the first time, the McCallum projection can be used and, in the absence of error messages, is proved to be valid. Some future implementation of CAD will doubtless allow for the addition of delineating polynomials (or, in the terminology of [3], allow for adding points to CAD's). Such an implementation would still benefit from the criteria developed in this paper.

6 Appendix A: Problems

This appendix describes some of the basic quantifier elimination problems considered in this paper.

6.1 The x-axis ellipse problem

The x-axis ellipse problem, a special case of the general ellipse problem posed by Kahan [10], is a traditional benchmark problem for quantifier elimination algorithms. (See for example [9], [8]) The problem asks when the ellipse $(x - c)^2/a^2 + y^2/b^2 = 1$ lies in the unit circle. Of course we require a and b to be non-zero, and in fact we are only interested in the case where they are positive. The formula

$$(\forall x)(\forall y)\left[\begin{array}{l} a > 0 \quad \wedge \quad b > 0 \quad \wedge \quad \left[b^2(x-c)^2 + a^2y^2 - \right. \\ a^2b^2 = 0 \longrightarrow x^2 + y^2 - 1 \leq 0\left.\right] \end{array}\right]$$

expresses this as a quantifier elimination problem.

6.2 Anai's problem

In [1], Hirokazu Anai applies the Virtual Term Substitution method of quantifier elimination [12] to problems in control theory. The particular problem we examine is the last example posed in the paper, which finally boils down to the following quantifier elimination problem:

$$(Ex_1)(Ex_2)\left[\begin{array}{l} s \geq 0 \wedge x_2 \geq 0 \wedge x_1 + 1 \geq 0 \wedge sx_2 - x_1^2 \geq 0 \\ \wedge x_2x_1 + x_2 \geq 0 \wedge s(x_1 + 1) \geq 0 \\ \wedge (x_2x_1 + x_2)s - x_1^3 - x_1^2 \geq 0 \wedge z - x_1 - x_2 \geq 0 \\ \wedge -10 \leq s \wedge s \leq 0 \end{array}\right]$$

.

6.3 The edge-square product problem

Consider the complex segment $L = \{x + i \mid x \in [0, 2]\}$, and the complex square $S = \{x + iy \mid x \in [2, 4], y \in [-1, 1]\}$. Quantifier elimination can be used to determine the complex product of S and L. Since there are the easily derived necessary conditions that the product lies in the box $[-1, 9] \times i[-6, 6]$, the prod-

uct can be expressed as all pairs (x, y) satisfying:

$$(\exists x_1)(\exists x_2)(\exists y_2)\left[\begin{array}{l} x = x_1x_2 - y_2 \quad \wedge \quad y = x_1y_2 + x_2 \quad \wedge \\ 0 \leq x_1 \leq 2 \quad \wedge \quad 2 \leq x_2 \leq 4 \quad \wedge \\ -1 \leq y_2 \leq 1 \\ \quad \wedge \quad -1 \leq x \leq 9 \quad \wedge \quad -6 \leq y \leq 6 \end{array}\right]$$

This problem appears in [4].

7 Appendix B: A true decision procedure for CONSTCOORDTEST

Here is a simple algorithm, "CONSTCOORDDECIDE", which takes a k-level cell c and a $(k+1)$-level projection factor P and determines whether or not the coordinates of the points in c are constant for all coordinates corresponding to variables appearing in P. It returns SUCCESS when this is the case, and FAILURE otherwise. The algorithm is impractically slow, but a fast, though weaker, variant of it is the previously discussed CONSTCOORDTEST.

CONSTCOORDDECIDE(k, c, P)

1. set $a = \{\}$ (note: a grows to be sort of a generalized sample point, where some coordinates are given values while others remain free, like $(x = -1, y = y, z = 0)$)
2. set $F = \{\}$
3. for i from 1 to k do
   (a) set c_i to the i-level ancestor of c
   (b) if c_i has dimension 0, add $x_i = \alpha_i$ to a, where α_i is the ith coordinate of the sample point for c_i
   (c) if c_i is a sector then
      i. if $deg_{x_i}(P) = 0$, add $x_i = x_i$ to a, otherwise return FAILURE
   (d) if c_i is a section cell of dimension greater than zero then
      i. set L to the set of i-level projection factors of which c_i is a section
      ii. set L' to the set of all elements of L "evaluated at a" (this is actually a partial evaluation, since some of the entries of a are of the form $x_j = x_j$), removing any that are identically zero
      iii. set L^* to the set of all elements of L' evaluated at $x_i = \alpha_i$, where α_i is the ith coordinate of the sample point for c_i

iv. if any element of L^* is in the ideal defined by the irreducible component of $V(F)$ that contains c_i (note that either all elements of L^* are in the radical of the ideal generated by the elements of F, or none are) then add $x_i = \alpha_i$ to a, otherwise

if $deg_{x_i}(P) = 0$ then set $F = F \cup L'$ and add $x_i = x_i$ to a, otherwise return FAILURE

4. return SUCCESS

As a simple example, let $P(x_1, x_2, x_3, x_4, z) = (x_1 - x_4 + 1)z^2 - x_1$ and let c be the 1-dimensional cell in $\mathbb{R}^4$ constructed via the following lifting steps:

	cell type	sections of ...	sample point coord.
Level 1	section	$\{x_1\}$	$\alpha_1 = 0$
Level 2	sector	—	$\alpha_2 = 3/2$
Level 3	section	$\{x_3 + x_2\}$	$\alpha_3 = -3/2$
Level 4	section	$\{x_3x_4 + x_2 + x_1\}$	$\alpha_4 = 1$

At the start of the 4th iteration of CONSTCOORDDECIDE, $F = \{x_3 + x_2\}$ and $a = \{x_1 = 0, x_2 = x_2, x_3 = x_3\}$. First L is set to $\{x_3x_4 + x_2 + x_1\}$, then L' is set to $\{x_3x_4 + x_2\}$, and finally L^* is set to $\{x_3 + x_2\}$. Clearly, the one and only element of L^* is in the ideal defined by the one and only component of $V(F)$, and therefore coordinate 4 of every point in c has the value 1.

In practice CONSTCOORDTEST seems to be reasonably successful at recognizing when a cell's coordinates are constant at each coordinate corresponding to a variable that appears in P. This can perhaps be explained as follows: The elements of F are polynomials in the variables that do not appear in P. The discriminant of P and other projection factors descended from P do not contain these variables, so if they appear in L, then the ideal membership from Step 3.d(iv) reduces to simple zero testing, which is precisely what CONSTCOORDTEST does.

References

[1] Anai, H. Solving LMI and BMI problems by quantifier elimination. In *Proc. of IMACS-ACA'98* (http://www-troja.fjfi.cvut.cz/aca98/proceedings.html, 1998), R. Liska, Ed. Electronic Proceedings.

[2] Arnon, D. S., Collins, G. E., and McCallum, S. Cylindrical algebraic decomposition I: The basic algorithm. *SIAM Journal on Computing 13*, 4 (1984), 865–877.

[3] Brown, C. W. Improved projection for cylindrical algebraic decomposition. *Journal of Symbolic Computation 32*, 5 (November 2001), 447–465.

[4] Brown, C. W. Simple CAD construction and its applications. *Journal of Symbolic Computation 31*, 5 (May 2001), 521–547.

[5] Caviness, B., and Johnson, J. R., Eds. *Quantifier Elimination and Cylindrical Algebraic Decomposition*. Texts and Monographs in Symbolic Computation. Springer-Verlag, 1998.

[6] Collins, G. E. Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition. In *Lecture Notes In Computer Science* (1975), vol. Vol. 33, Springer-Verlag, Berlin, pp. 134–183. Reprinted in [5].

[7] Collins, G. E., and Hong, H. Partial cylindrical algebraic decomposition for quantifier elimination. *Journal of Symbolic Computation 12*, 3 (Sep 1991), 299–328.

[8] Dolzmann, A., and Sturm, T. Simplification of quantifier-free formulae over ordered fields. *Journal of Symbolic Computation 24*, 2 (Aug. 1997), 209–231. Special Issue on Applications of Quantifier Elimination.

[9] Hong, H. Simple solution formula construction in cylindrical algebraic decomposition based quantifier elimination. In *Proc. International Symposium on Symbolic and Algebraic Computation* (1992), pp. 177–188.

[10] Kahan, W. Problem no. 9: An ellipse problem. *SIGSAM Bulletin of the Assoc. Comp. Mach. 9*, 35 (1975), 11.

[11] McCallum, S. An improved projection operator for cylindrical algebraic decomposition. In *Quantifier Elimination and Cylindrical Algebraic Decomposition* (1998), B. Caviness and J. Johnson, Eds., Texts and Monographs in Symbolic Computation, Springer-Verlag, Vienna.

[12] Weispfenning, V. Quantifier elimination for real algebra — the quadratic case and beyond. *AAECC 8* (1997), 85–101.